Dtrace Initial Prototype#

Our initial prototype is based on the idea of seeing what we can do. There are many dirty hacks - these will be cleaned up in the slower, more careful port going on in the trunk. In the meantime, this is a good place to get your feet wet and try to crash your kernel.

Get the source#

The source for the prototype lives in it's own branch (branches/old_dev_dtrace) - you can find it here

Build the source#

You will need to setup a staging directory. See this page for guidelines on how to do this.
Right now it's not recommend you share a staging dir with the coreos project, since the new elf headers probably won't work quite right.
Then - just type

make CPULIST=x86 hinstall
make CPULIST=x86 install

(note that only x86 works right now). If all goes well you should end up with

stage/x86/sbin/io-dtrace
stage/x86/bin/dtrace
stage/x86/lib/libdtrace.so.1

Run it!#

This prototype actually runs the dtrace code in a resource manager. It's quaintly named io-dtrace. This needs to be run before the dtrace utility will actually do anything much. You can list the available probes with

./x86/sbin/io-dtrace &
./x86/bin/dtrace -l

Try this one

```
# dtrace -n "kercalls::enter { @count[execname,probefunc] = count(); }"
```

and then wait a while and hit ctrl-c